# Computational Semantics

## Assignment 4: Compositional Semantics

## 1. Spatial relations

See images directory in .zip file.

## 2. Exercise 2.3.1

Vincent: λu.(u@VINCENT)
Mia: λy.(y@MIA)

Loves: λw.λz.(w@λx.LOVE(z,x))

—> Functional application LOVES, MIA

λw.λz.(w@λx.LOVE(z,x))@λy.(y@MIA)

—> β-conversion λw

λz.(λy.(y@MIA)@λx.LOVE(z,x))

—> β-conversion λy

λz.(λx.LOVE(z,x)@MIA))

—> β-conversion λx

λz.(LOVE(z,MIA))

—> Function application LOVES MIA, VINCENT

λu.(u@VINCENT))@λz.LOVE(z,MIA)

—> β-conversion λu

λz.(LOVE(z,MIA)@VINCENT)

—> β-conversion λz

LOVE(VINCENT,MIA)

## 3. Compositional semantics DCG grammar

```
n([tree:n(owl),sem:lam(X,n_owl_1(X))]) --> [owl].
n([tree:n(eye),sem:lam(X,n_eye_1(X))]) --> [eye].
n([tree:n(desert),sem:lam(X,n_desert_1(X))]) --> [desert].
tv([tree:tv(standing),sem:lam(P,lam(X,app(P,lam(Y,s_supports(Y,X))))))]) --> [standing,in].
iv([tree:iv(looking),sem:lam(X,some(Y,and(n_eye_1(Y),and(s_part_of(Y,X),a_open_1(Y))))))]) --> [looking].
mod([tree:mod(in),sem:lam(X,some(Y,and(n_owl_1(Y),and(s_supports(Y,X))))))]) --> [in].
mod([tree:mod(at),sem:lam(X,some(Y,and(n_eye_1(Y),and(s_part_of(Y,X),a_open_1(Y),a_looking_1(Y))))))]) --> [at].
```